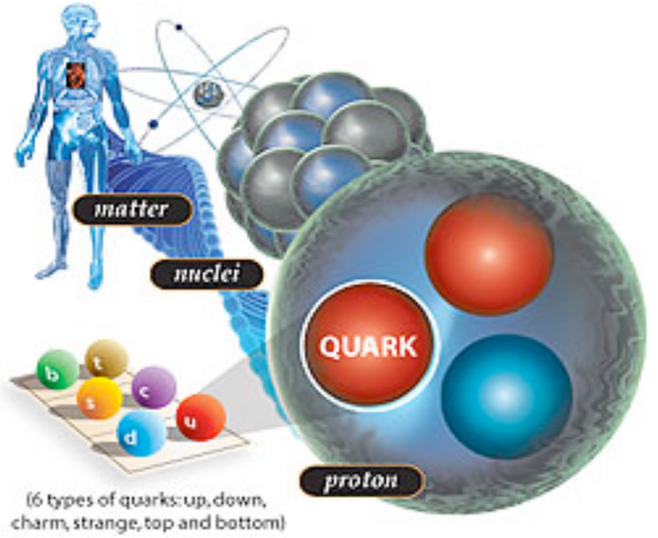
# Lattice QCD for Nuclear Physics

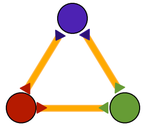
PI: Robert G. Edwards (Jefferson Lab)
Co-PIs:
Will Detmold (MIT), **Bálint Joó (Jefferson Lab)**, Swagato Mukherjee (BNL)
SciDAC FastMATH Institute workshop, ANL
June 10, 2019 (Remote presentation)

# Introduction

- It is believed that the fundamental building blocks of matter are ***quarks*** bound together by ***gluons***, via the ***strong nuclear force.***
- Quantum Chromodynamics (QCD) is the theory which describes the strong interactions
- Understanding how QCD makes up matter and how quarks and gluons behave is a subject of intense experimental scrutiny
  - only ~5% of the mass of a proton comes from mass of the quarks, rest comes from binding
  - gluon self-coupling and gluon excitations can create ***exotic forms of matter***

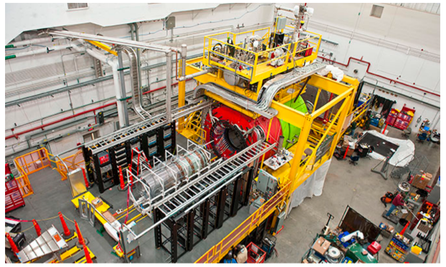



meson: 2 quarks



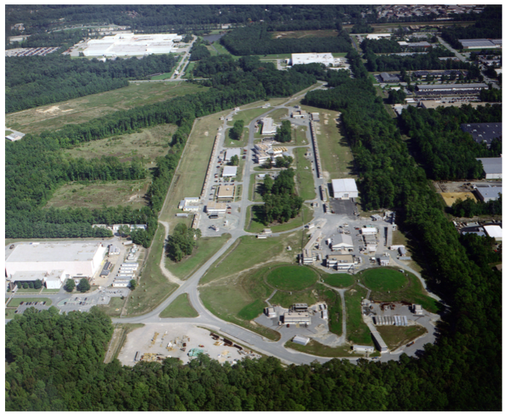
baryon: 3 quarks

glueball: 0 quarks only gluons



***GlueX in the new Hall-D of Jefferson Lab@12 GeV. Hunting for exotics!***

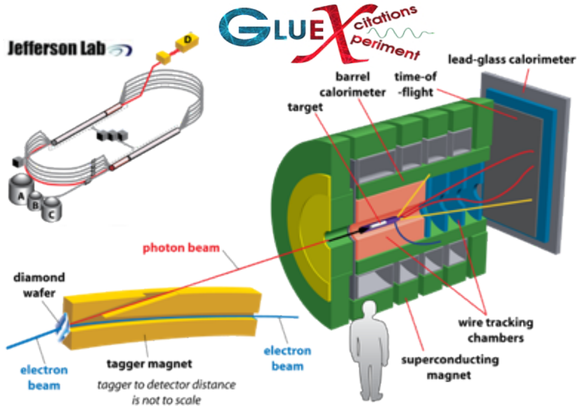
***Jefferson Lab***

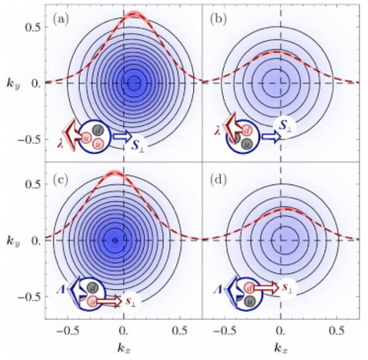


***Brookhaven National Lab***

# Important questions in Nuclear Physics

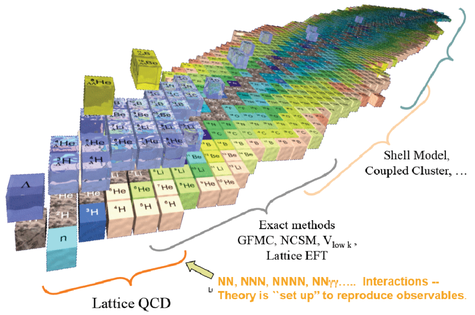
- What observable states does QCD allow?
  - what is the role of the gluons?
  - what about exotic matter?



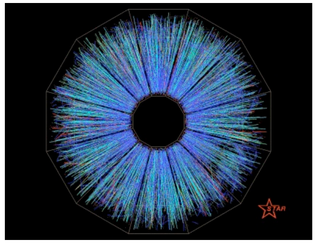
Hägler, Musch, Negele, Schäfer, EPL 88 61001

- How does QCD make protons, neutrons?
  - what are the distribution of quarks, gluons, etc in a proton or neutron ?




- QCD must predict properties of light nuclei
  - how to make helium, tritium etc

- How does QCD behave under extreme temperatures & pressures such as in supernovae or shortly after the Big-Bang.



# QCD: Path Integral Formulation

- Quarks and Gluons are ‘fields’ in space-time (Minkowski Space)
- QCD Is defined by the Action (S) over the fields
- Action enumerates potential interactions
  - quark-gluon, gluon-gluon etc.
- **Observables** can be computed through Path Integrals over the fields.
- Running Coupling (Asymptotic Freedom)
  - Large energies => Perturbation Theory
  - Low energy => Nonperturbative Method needed.

“Functional Integral” over all the possible states of the fields

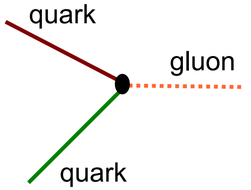
The “action” defining the theory

$$\langle \mathcal{O} \rangle = \frac{1}{\mathcal{Z}} \int \mathcal{D}A \, \mathcal{D}\bar{\psi} \, \mathcal{D}\psi \, \mathcal{O} \, e^{-S(A,\bar{\psi},\psi)}$$

Expectation value of an observable (correlation function)

Value of observable on a concrete set of fields

$$S = \int dx \; dy \; \bar{\psi}(y) M(A; y, x) \psi(x)$$




$$-\int dx \; \frac{1}{4} G_a^{\mu\nu}(x) G^a_{\mu\nu}(x)$$

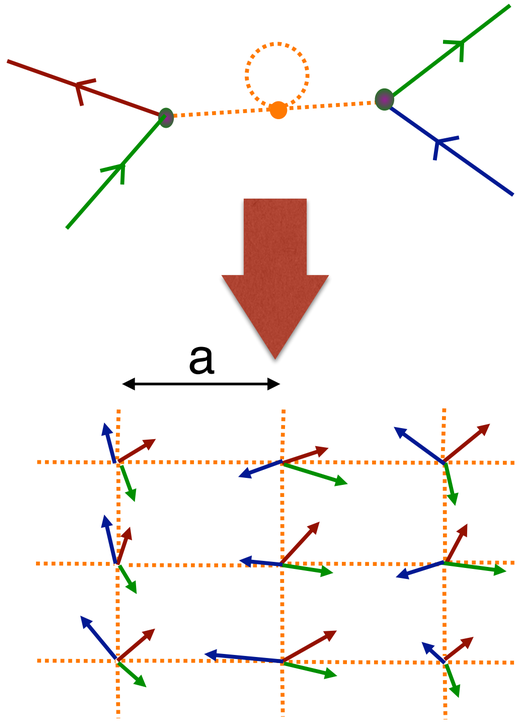
# Lattice QCD

- Replace “continuum” space time by 4D Lattice
  - 3D spatial box (finite volume)
  - Euclidean Time ( t => it ) ***$L_t$ is proportional to 1/Temperature***
- Discretize quark fields onto lattice sites
- Discretize gluon fields onto lattice links as SU(3) matrices

$$U_\mu(x) = \exp\left\{ ig \int_x^{x+\hat{\mu}} dx\ A_\mu(x) \right\} \approx \exp\{ig\ a\ A_\mu(x)\}$$

  - 3x3 matrices on links act as “parallel transporters” along links
- Finite differences for derivatives
- Functional integrals become ‘regular’ integrals
- QCD is continuum limit of LQCD (RG)
- LQCD is not a ``model”: Fully renormalizable QFT!!!

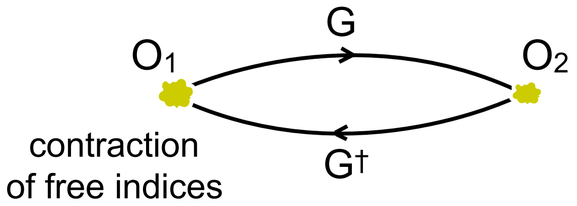
$$\langle \mathcal{O} \rangle = \frac{1}{\mathcal{Z}} \int \mathcal{D}A\ \mathcal{D}\bar{\psi}\ \mathcal{D}\psi\ \mathcal{O}\ e^{-S(A,\bar{\psi},\psi)}$$




$$\langle \mathcal{O} \rangle = \frac{1}{\mathcal{Z}} \int \prod_{\text{all links}} dU \prod_{\text{all sites}} d[\bar{\psi},\psi]\ \mathcal{O}\ e^{-S(U,\bar{\psi},\psi)}$$

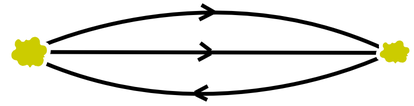
# What are our observables?

- Lattice QCD Observables are “correlation functions”
- E.g. For mesons (quark-antiquark pairings):




Meson: e.g. the π meson (a.k.a pion)

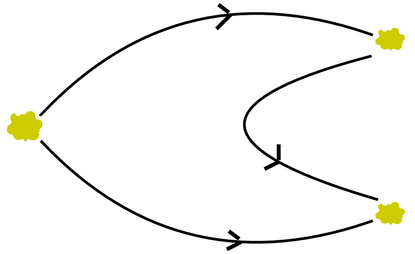
$$C(\vec{p},t) = \sum e^{i\vec{p}.\vec{x}}\ \mathrm{Tr}\ \{O_1\ G_1^\dagger\ O_2\ G_2\} = \sum A_i\ e^{-\sqrt{E_i^2+|\vec{p}|^2}t} \xrightarrow[t\to\infty]{} A_0\ e^{-\sqrt{m^2+|\vec{p}^2|}t}$$



Baryon: e.g. proton or neutron

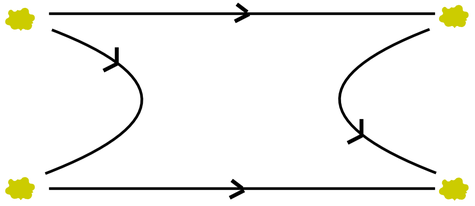
- G is the quark propagator defined as:

$$G(x,y) = M^{-1}_{x,y} S(x)$$



Meson to 2 meson decay: e.g. ρ → 2π

- M is the Fermion matrix
- Computing C(t) involves ***solving linear systems for G*** (Solvers) and ***contractions (few index)***



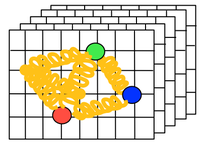
2 meson in to 2 meson out: e.g. 2π → 2π

# The name of the game:

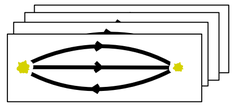
- ***Produce Ensembles of Configurations***
  - Several box-sizes for finite V behavior
    - E.g. Luscher formalism for phase shifts, resonances etc.
    - Several Temperatures (in Finite temperature calculations)
  - Several lattice spacings ‘a’ (to allow for continuum limit extraction)
  - As high statistics as possible (sample path integral well)
    - ***Want to minimize time for sample generation, minimize autocorrelation time (algorithms)***
- ***Evaluate Correlation Functions***
  - Need ***O(100,000)-O(1M)*** quark propagators per configuration (reduce signal to noise)
    - ***Want the fastest possible multi-RHS linear solvers***
  - Can have ***O(10,000)-O(100,000)*** quark line diagrams for some observables
    - ***Want to eliminate as many diagrams as possible using symmetry***
    - ***Want fast I/O for database accesses***
    - ***Want to maximize common by-product reuse***
    - ***Want automation & job orchestration: Workflows!!!!***
- ***Want to run on any hardware that falls into our lap:*** ***Performance portability Strategy***

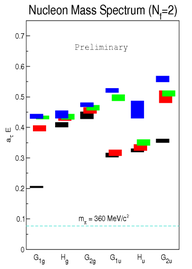
# Anatomy of an LQCD calculation


Gauge Configurations




Propagators, Correlation Functions






Gauge Generation: {U}

Measurement: O(U)

Fitting

Physics Result

- Gauge Generation: create snapshots of the LQCD Vacuum
  - sequential (Markov Chain) but has data parallelism from the lattice
  - strong scaling limited: needs leadership computing resources
- Measurement:
  - computes propagation of quarks on the gauge field snapshots
  - contracts propagators into correlation functions
  - throughput limited: very cost effective on clusters like at JLab or ensemble mode running at LCFs

*Titan Image Courtecy of OLCF*
*Edison Image Courtesy of NERSC*

# Importance Sampling

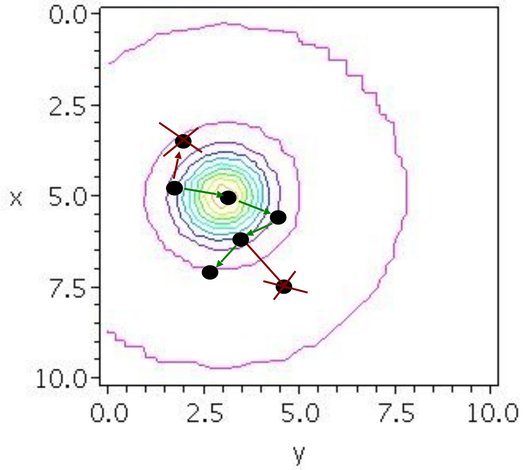
- Pick Configuration ‘U’ with probability P(U)
- Ensemble average then becomes a ‘regular average’

$$\langle \mathcal{O} \rangle = \frac{1}{\mathcal{Z}} \int \prod_{\text{all links}} dU_i \; \mathcal{O} \; e^{-S(U)} \longrightarrow \bar{O} = \frac{1}{N} \sum_{\text{N}} \mathcal{O}(U) \qquad \sigma(\bar{\mathcal{O}}) \propto \frac{1}{\sqrt{N}}$$

- E.g.: Metropolis Algorithm
  - Start from some initial configuration U
  - Pick trial config U’ from U **reversibly:** i.e. $P_c(U \rightarrow U') = P_c(U' \rightarrow U)$
  - Accept with Metropolis probability

$$P(U' \leftarrow U) = \min\left(1, \frac{e^{-S(U')}}{e^{-S(U)}}\right)$$

  - If we reject, next config is U again



# Hybrid Monte Carlo

- Big Trick: Update all links at once using Molecular Dynamics
  - Treat each link as ‘canonical coordinate’
  - Assign to each link a ‘canonical momntum’ in the lie algebra su(3)
- Construct a fictitious Hamiltonian

$$H = \frac{1}{2}\sum_{\text{links}} p^2 + S(U)$$

- Simulate Hamiltonian System with partition function:

$$\mathcal{Z} = \int \mathcal{D}U\ \mathcal{D}p\ e^{-H} = \int \mathcal{D}U\ e^{-S} \int \mathcal{D}p\ e^{-\frac{1}{2}\sum_{\text{links}} p^2} = C \int \mathcal{D}U e^{-S(U)}$$

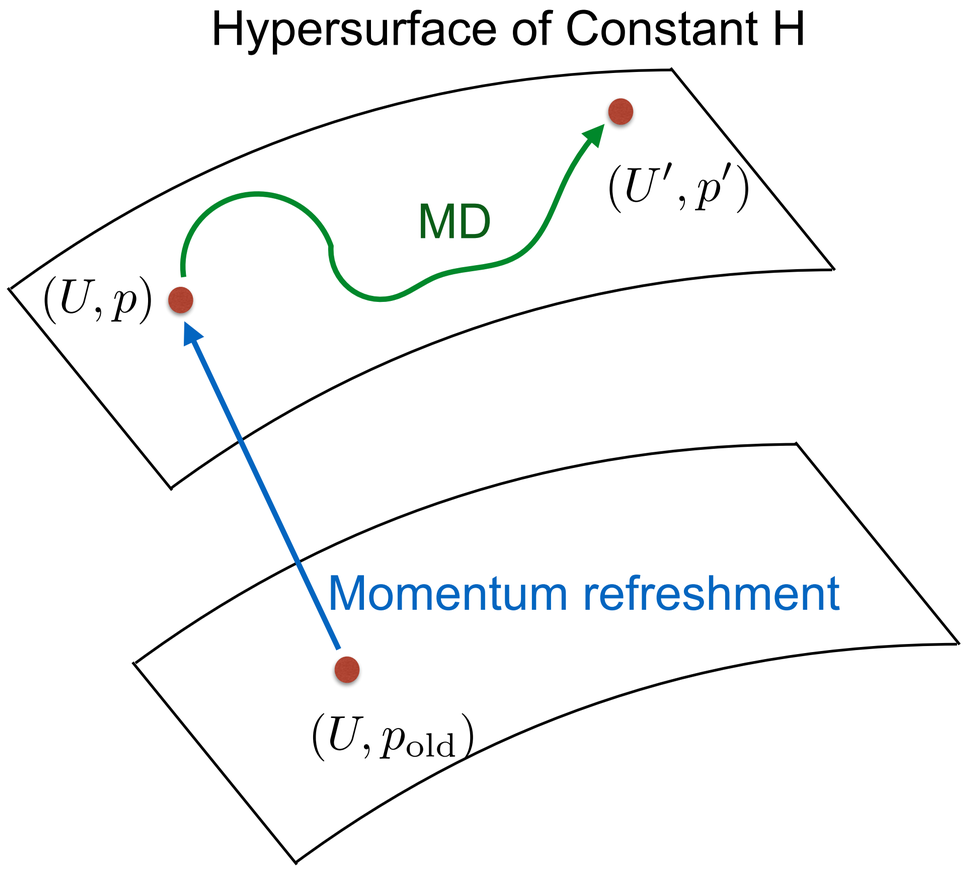
- Momenta have gaussian distribution: easy to generate from heatbath
- Keep the Marginal Distribution of the gauge fields (ignore momenta)

# Hybrid Monte Carlo (HMC)

1. Refresh momenta from Gaussian Heatbath
   - generate (U,p) from (U,$p_{old}$)
2. Compute H = H(U,p)
3. Perform Molecular Dynamics trajectory
   - generate (U',p')
   - MD must be reversible and 'area/measure preserving'
     - To Guarantee Detailed Balance
4. Compute H' = H(U',p')
5. Accept with Metropolis probability

$$P = \min\left(1, e^{-H(U',p')+H(U,p)}\right)$$

6. If (U',p') is rejected, the new state is (U,p)



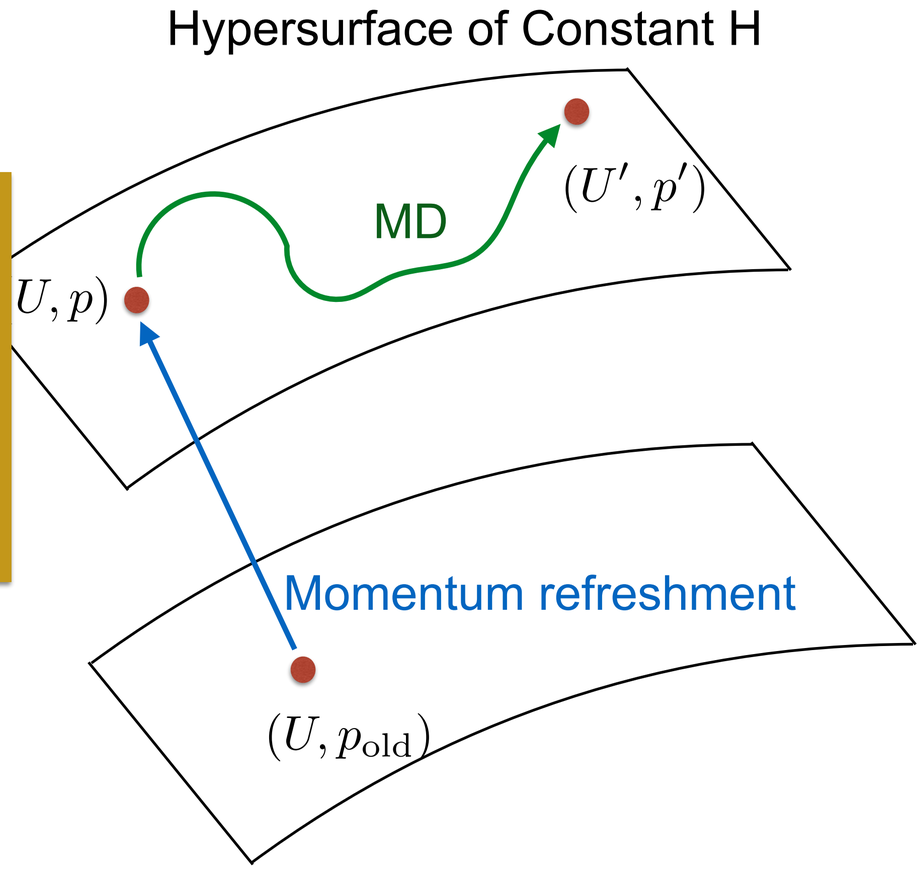
# Hybrid Monte Carlo (HMC)

1. Refresh momenta from Gaussian Heatbath
   - generate (U,p) from (U,$p_{old}$)
2. Compute H = H(U,p)
3. Perform Molecular Dynamics
   - generate (U',p')
   - MD must be reversible and 'are
     - To Guarantee Detailed Balan
4. Compute H' = H(U',p')
5. Accept with Metropolis probability

$$P = \min\left(1, e^{-H(U',p')+H(U,p)}\right)$$

6. If rejected new state is (U,p)

About 95% of time is spent in MD Force Calculations



# MD Forces

- Momentum Update: $e^{\delta\tau \hat{P}} : \; p(\tau + \delta\tau) \rightarrow p(\tau) + \delta\tau F$
- For 2 Flavor Quark Action:

$$F = -\phi^\dagger \left(M^\dagger M\right)^{-1} \left[\dot{M}^\dagger M + M^\dagger \dot{M}\right] \left(M^\dagger M\right)^{-1} \phi$$

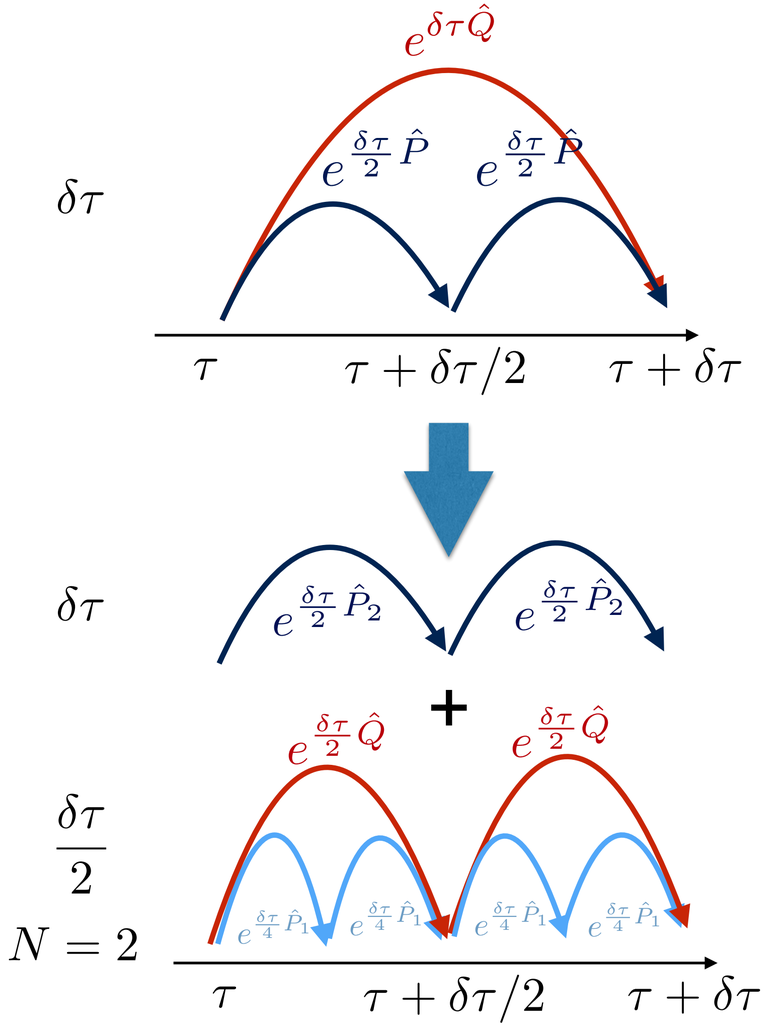
- Need to evaluate:

$$\left(M^\dagger M\right) X = \phi$$

- Here again we need a solver but:
  - System is manifestly Hermitian and Positive definite
  - Common to use two step solve: $M^\dagger Y = \phi$ followed by $M X = Y$ (reduced condition number)
  - M will change as we perform the MD gauge field update, long set-up times for solver may not be as easily amortized as for propagators.

# State Of The Art

- Hybrid Monte Carlo Algorithm
  - Determinant Splitting.
  - **Multiple Time-scale integration using Force-Gradient Term.**
  - Chronological Solution Predictors.
  - **Aggregation Multi-Grid Solver**
- Propagator Calculations
  - **Aggregation Multi-Grid Solver**
- Contractions/Analysis - this varies from calculation to calculation
  - Spectroscopy:  Distillation Method



# Adaptive Multigrid in LQCD

- Critical Slowing down with decreasing quark mass is caused by ‘near zero’ modes of M
- Multi-Grid method based on Adaptive Smoothed Aggregation
  - separate (project) low lying and high lying modes
  - reduce error from high lying modes with “smoother”
  - reduce error from low modes on coarse grid
  - Gauge field is ‘stochastic’, so no geometric smoothess on low modes => algebraic multigrid
  - Setting up restriction/prolongation operators has a cost
  - Easily amortized in Analysis with O(100,000) solves
- ***QCD Version thanks to SciDAC 1,2,3!***
  - Collaboration between R. Falgout & J. Brannick with Boston University Group (R. Brower, K. Clark, R. Babich, C. Rebbi, J. Osborn and others)

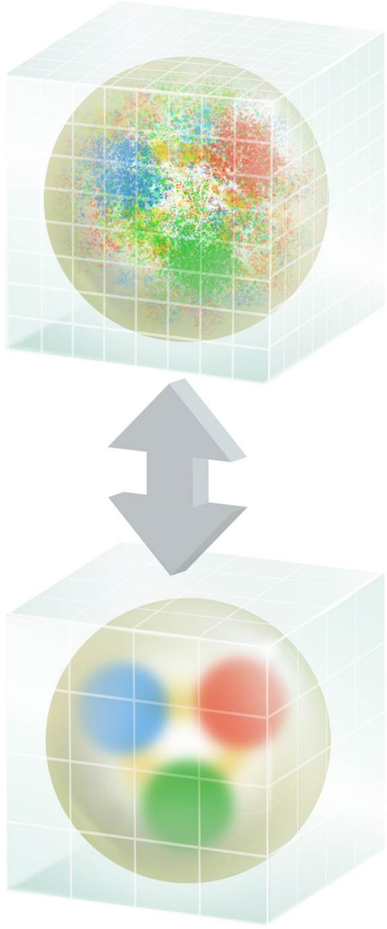
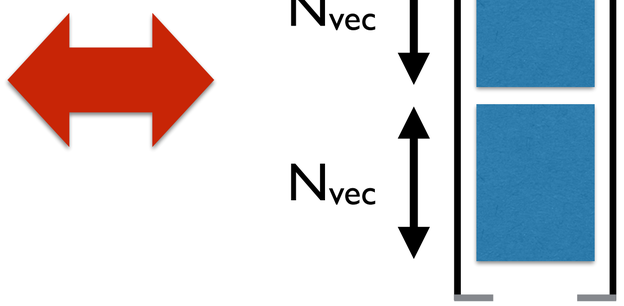


Image Credit: Joanna Griffin, Jefferson Lab Public Affairs

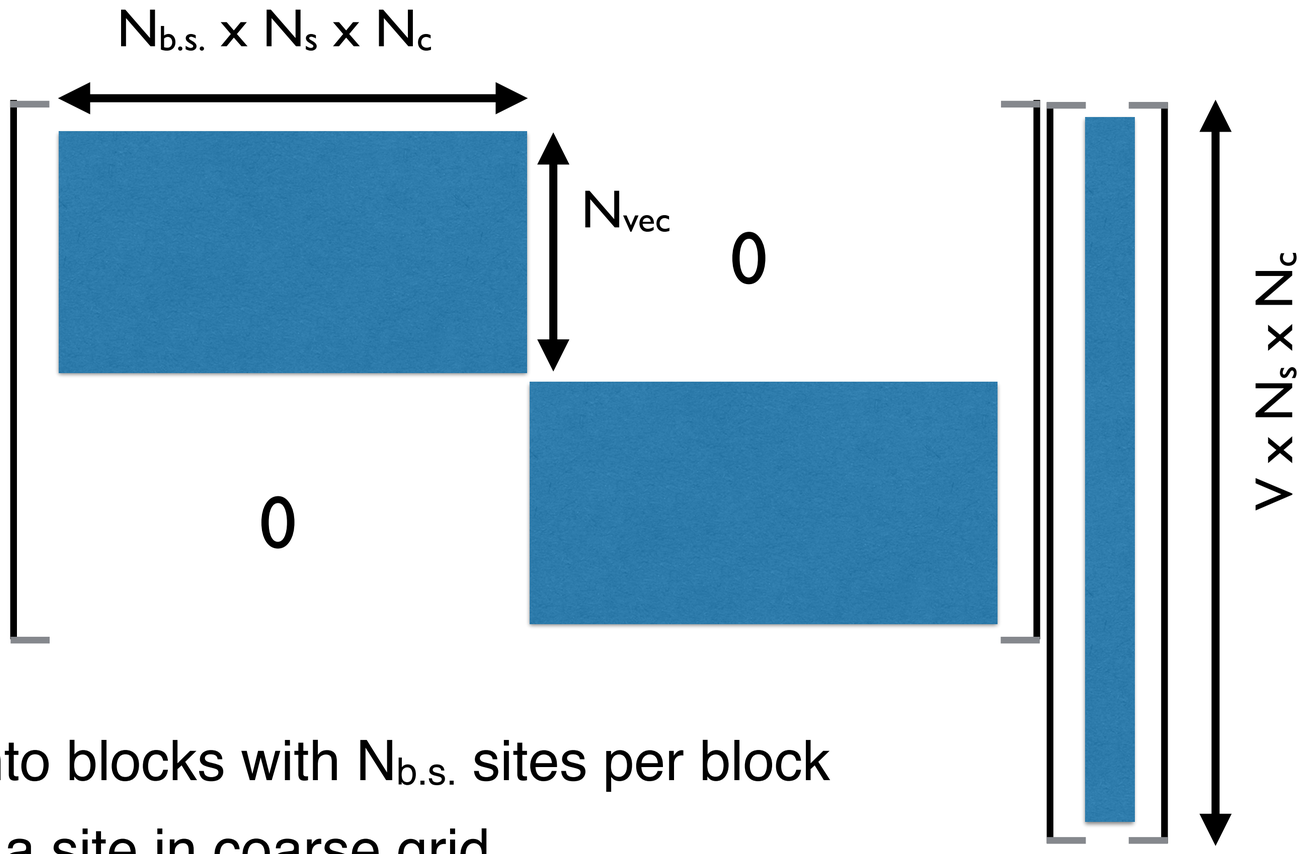
# Block Aggregation

## Restriction via aggregation:

$v_c = R\, v_f$



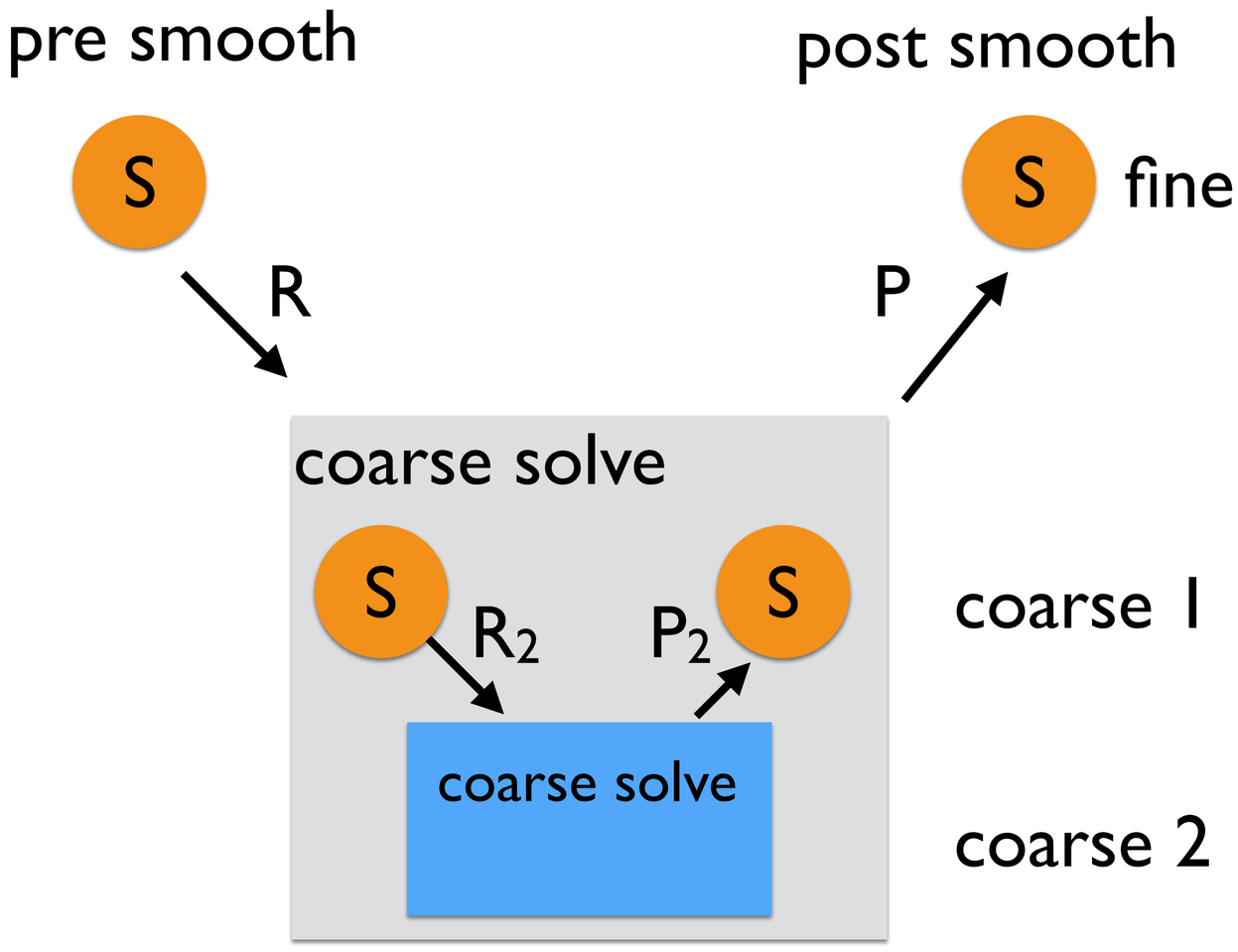

=




- Generate $N_{vec}$ near-nullspace vectors, 'cut' into blocks with $N_{b.s.}$ sites per block
- Aggregate over blocks: each block becomes a site in coarse grid
- $N_{vec}$ becomes number of 'colors' on each coarse grid site
- Typically use $P = R^{\dagger}$ and Galerkin Coarse Operator $M_c = R\, M\, P$

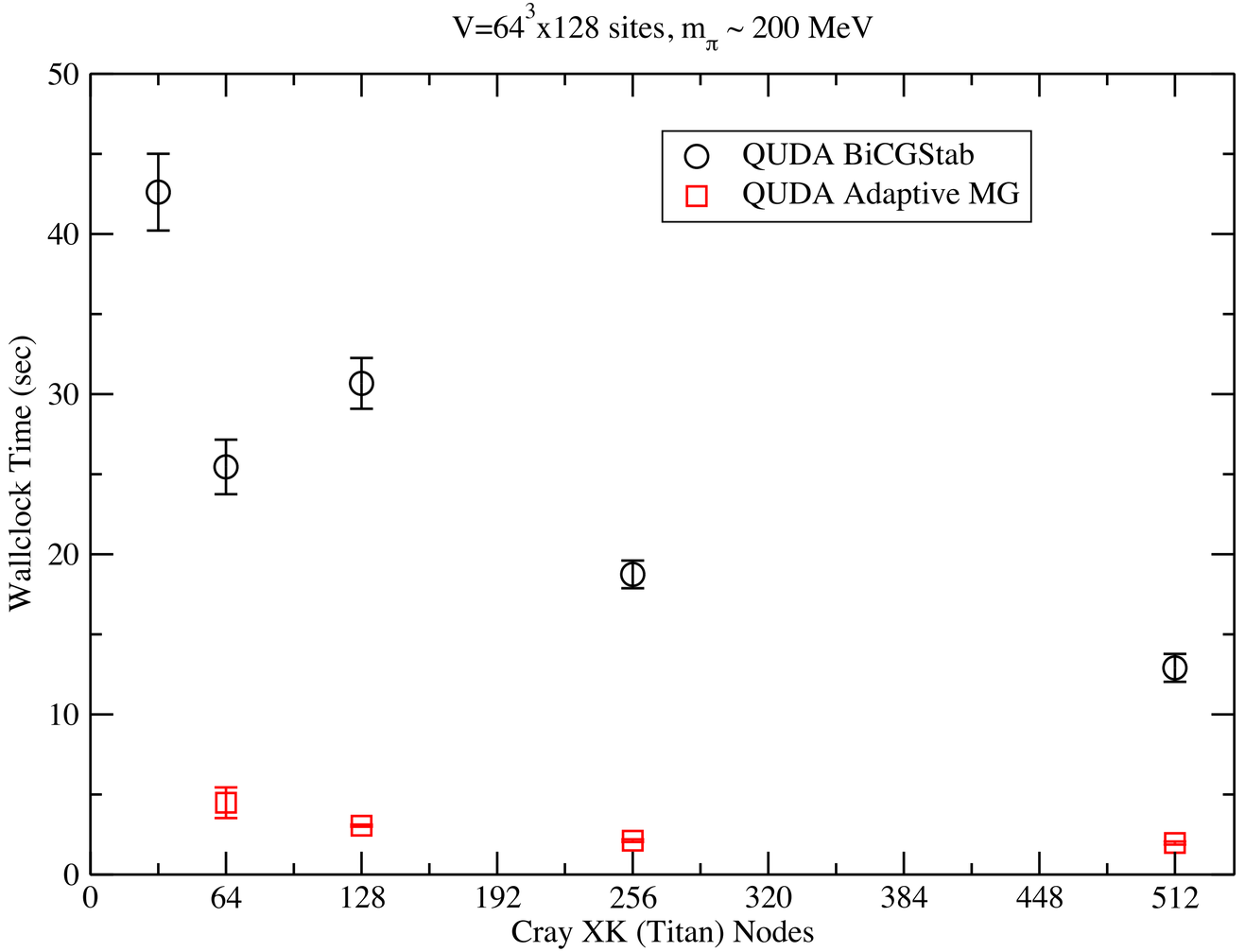
# Typical Implementation

- Outer Flexible Krylov Method
  - e.g. GCR, FGMRES
- MG V-cycle used as a Preconditioner.
  - Null space:
    - Solve M x = 0 for $N_{vec}$ random x, construct R, P
    - more elaborate schemes - research topic
  - Smoother: MR or Block Jacobi
  - 'Bottom Solver':
    - GCR/FGMRES
    - May be deflated (e.g. FGMRES-DR, or explicitly)
    - May be recursively preconditioned (e.g. by MG)
- Very large space of algorithmic combinations/ tuning possibilities: MG cycles, parameters etc.



# Benefits of Multigrid: Speed

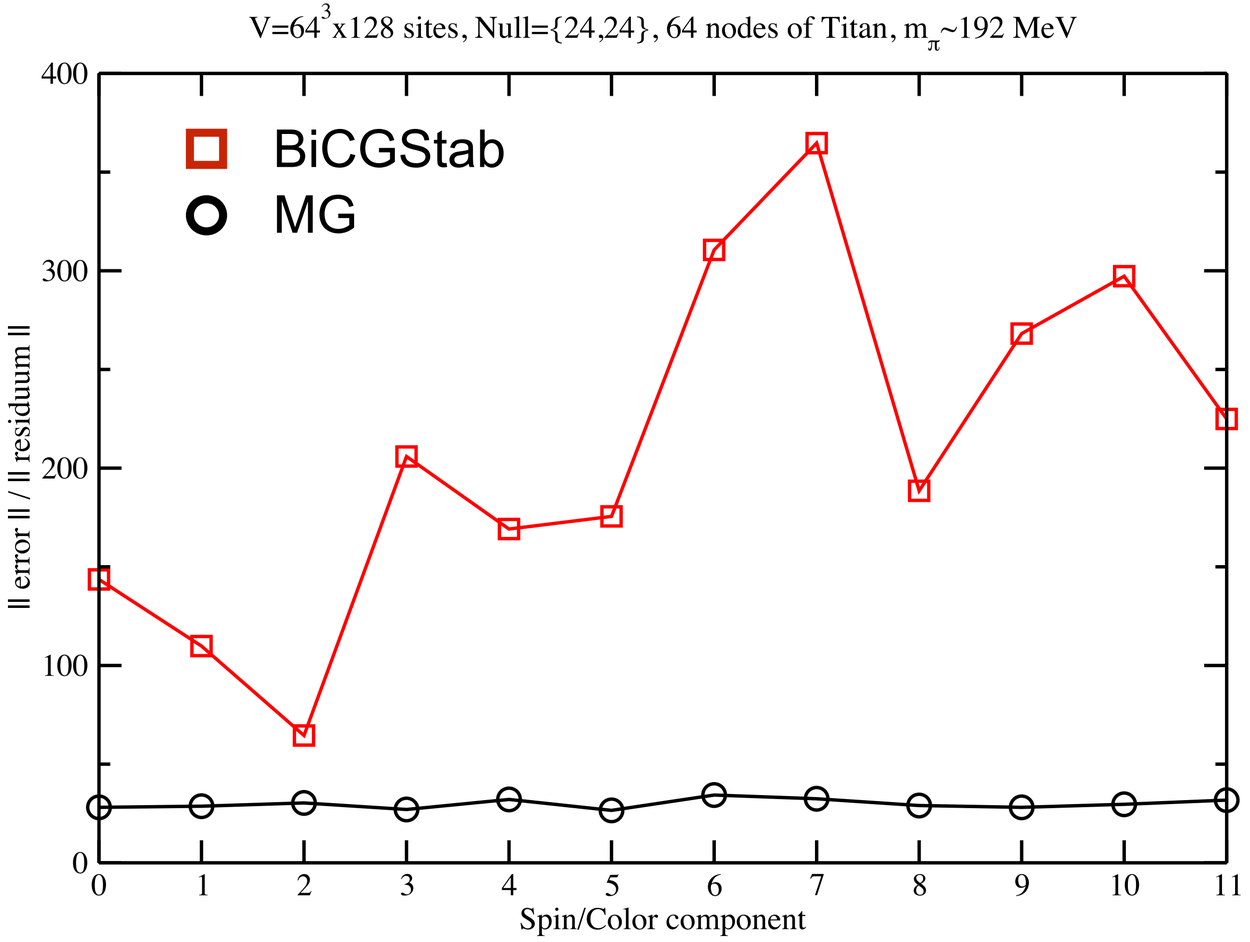
- Coarse Grids capture the troublesome low modes, but with massively reduced number of degrees of freedom.
- Typical Grid Blockings:
  - Level 1: 4x4x4x4 = 256x
  - Level 2: 2x2x2x2 = 16x
  - Overall: 4096x reduction
- MG is a preconditioner
  - reduced precision can be used beneficially without hurting precision of end result
- State of the art GPU implementation in QUDA Library (Clark et. al.)
- Not unreasonable x86 multicore implementation in mg_proto library (Joo)




from K. Clark et. al. SC'16

# Benfits of Multigrid: Optimality

- MG minimizes error, rather than residuum
- Solver is better behaved than BiCGStab
- number of iterations is stable
- ll error ll/ll residuum ll is more stable
- Important for t-to-same-t propagators
  - single precision is good enough BUT:
  - want precision guarantee from solve to solve



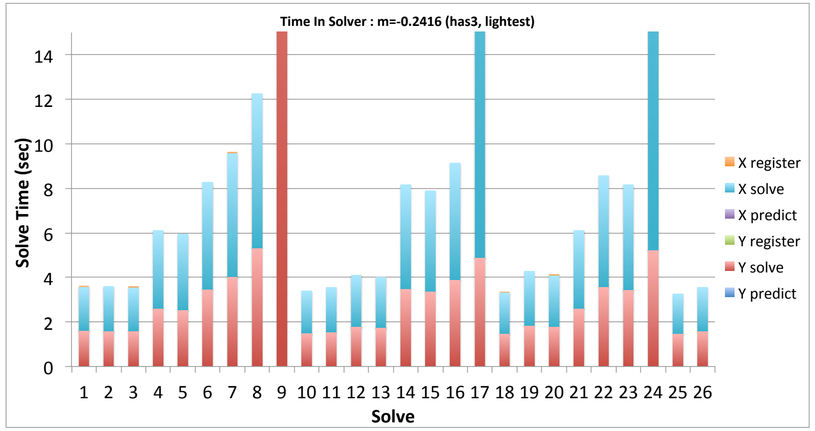

from K. Clark et. al. SC'16 - sneak preview

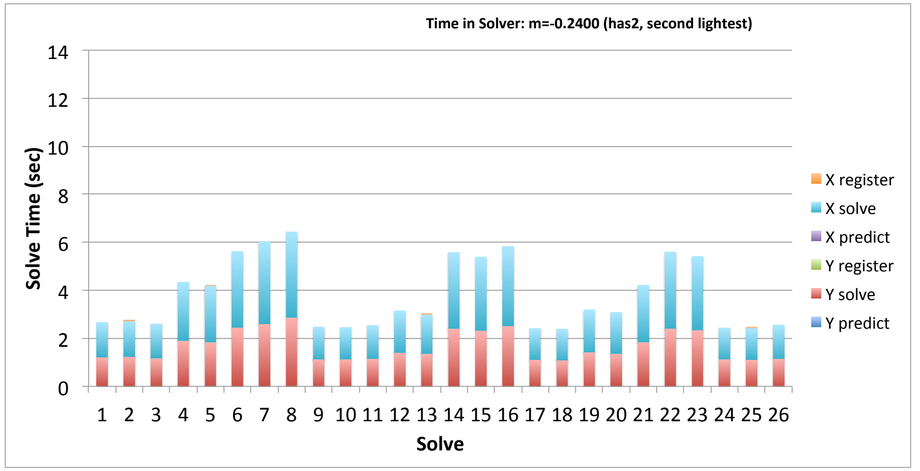
# Multigrid in the HMC

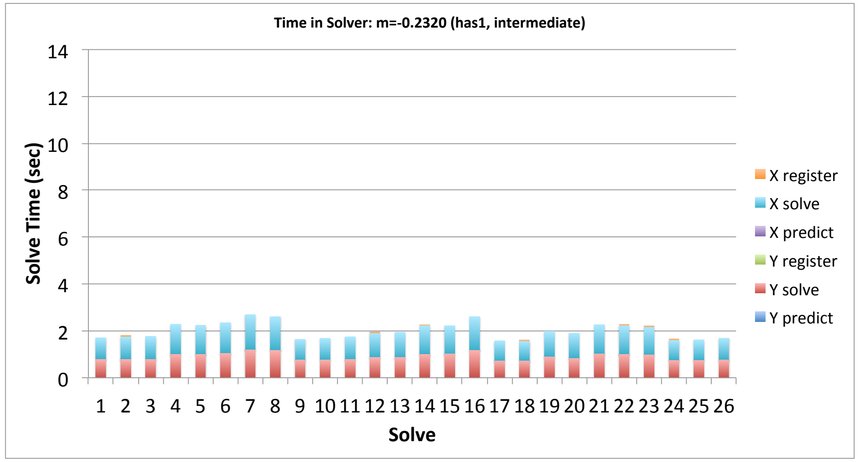
- We focus on the MG implementation in QUDA hooked into Chroma
- Primary Targets: Titan, Summit, Blue Waters and any other big GPU based system we can use
- MG Subspace Management
  - Keep Subspace in Chroma Named Object Store (reuse for solves with several masses)
  - Still need to recompute coarse operators: this must be fast
  - Subspaces lose effectiveness as fields evolve => Refresh when Iteration Threshold reached
  - Keep costs of Subspace Refresh low: use fixed number of iterations (rather than residuum)
  - Two new tunable parameters: Iteration Threshold & Refresh Iterations
- Numerical Experiments: 64^3x128 lattice (a~0.092fm, mpi ~ 172 MeV)
  - Production Isotropic Lattice

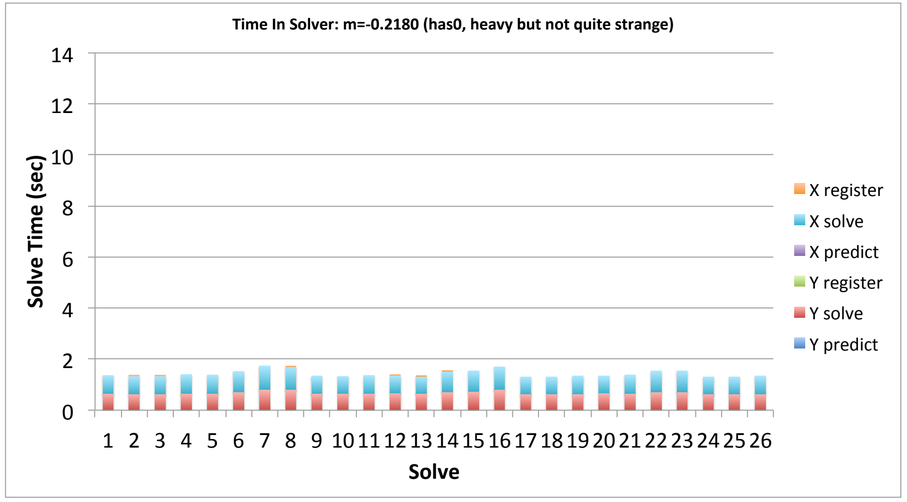
# Multigrid Behavior

- Only the lightest mass really matters — as expected. And as shown before by M. Lin the preconditioner doesn't degrade much for heavier masses

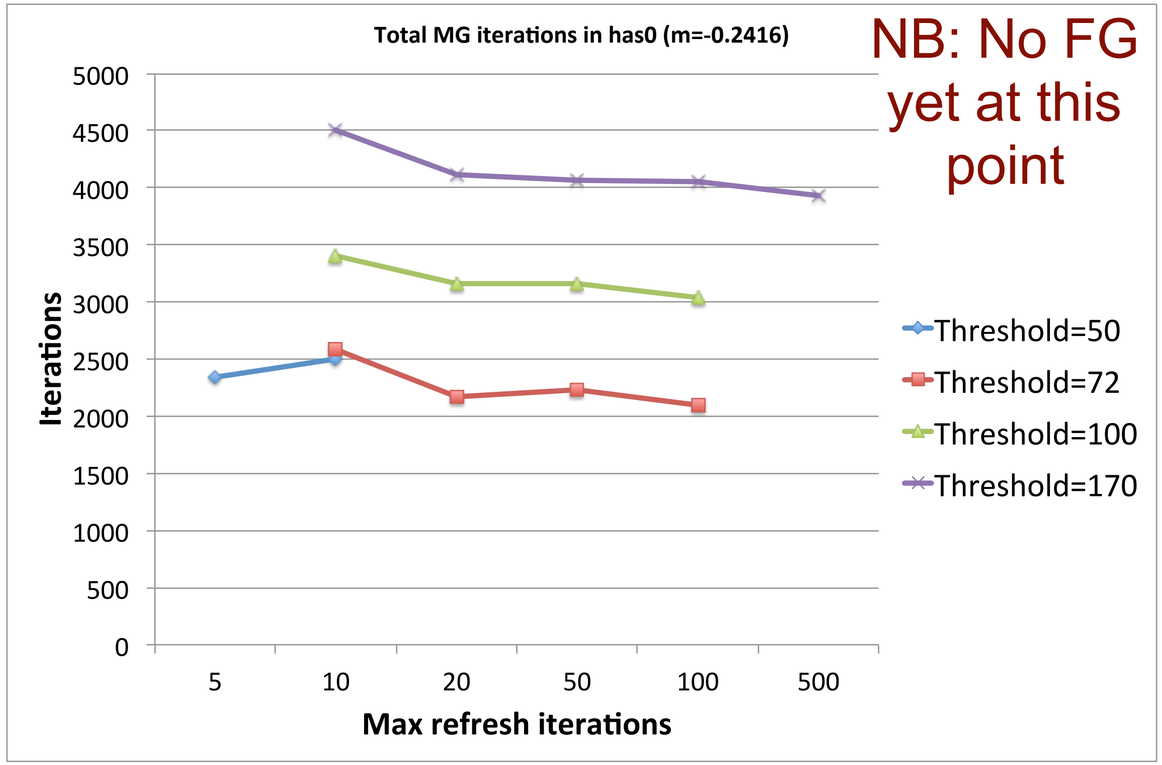


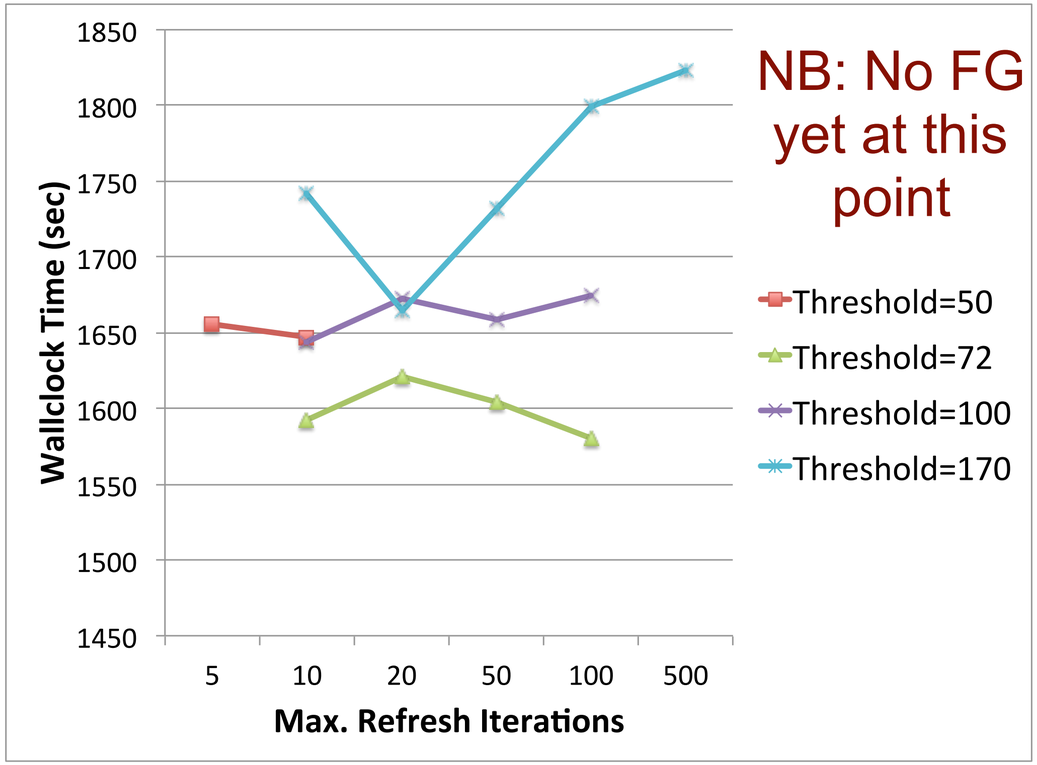



# Multigrid Tuning

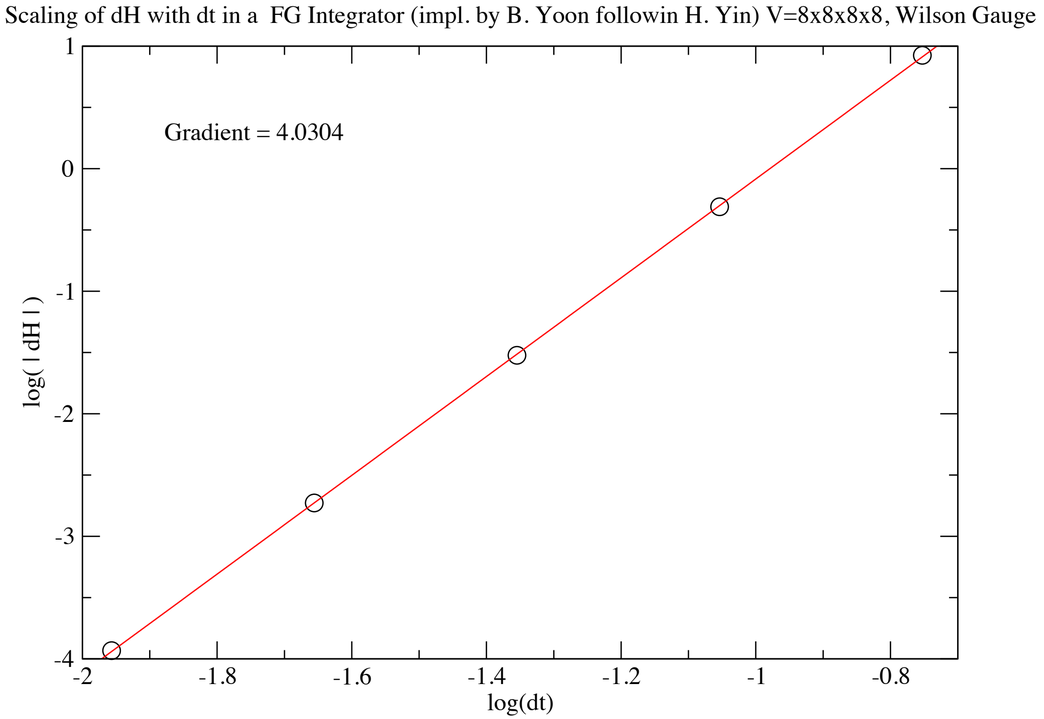


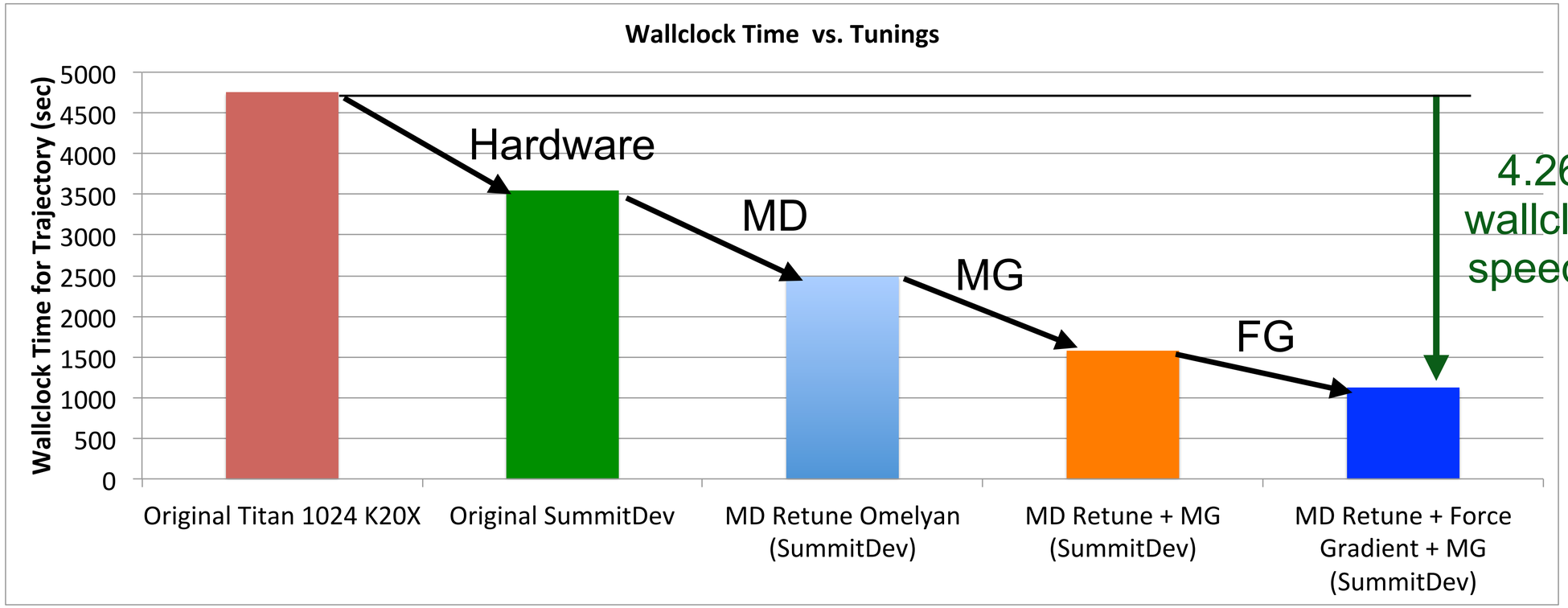

- Outer MG iterations insensitive to refresh iterations once refresh iters > 20 or so
- In terms of time, it is not immediately clear where the point is…
  - For Threshold=170 it is what I expect: there is a nice minimum (fewer iters=>worse space, more iters=> expensive)
  - For Threshold=72, we refresh sufficiently often to keep a low iteration count?

# Force Gradient Integrator

- Standard 4th order integrator following Omelyan requires 5 force evaluations per step (4MN5FV version)
- Omelyan 2nd order intergrator requires 3 force evaluations per step
- Force gradient integrator (Clark, Kennedy, Silva) following H. Yin and Mawhinney's exponential trick needs 3 force evaluations + 1 auxiliary force gradient evaluation, but is 4th order
  - Saves on solves compared to 4MN5FV
  - 4th order so volume scaling of cost is $V^{9/8}$.



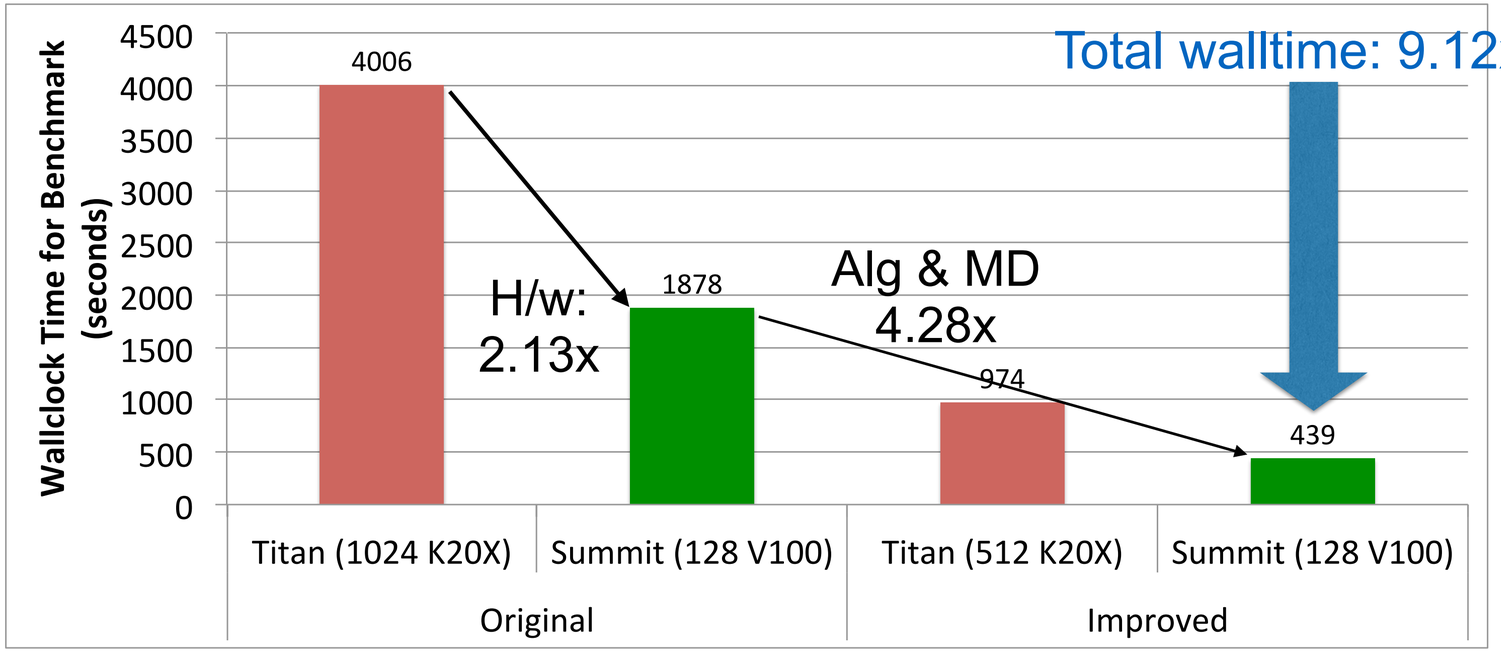
# Summit Dev Gains.




- 4.25x wallclock speedup on SummitDev
- On Titan, after all algorithmic optimizations on 512 nodes: ~974 sec per traj

# Summit Gains

*SciDAC Highlight In 2018!*




*Can We Get To 100x ?*

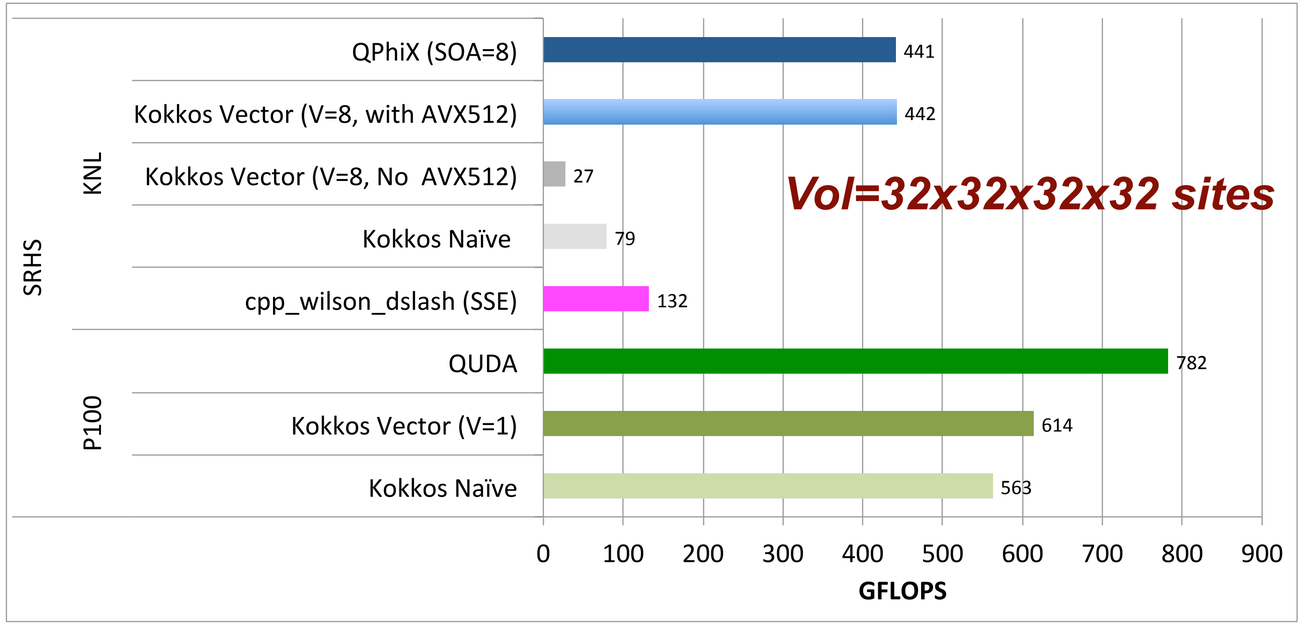
- Combined Hardware x Algorithm & MD Retuning Walltime gain: **9.12x**
- 8x reduction in GPUs: Integrated gain: **73x**. (Contribution to 2018 ECP FOM)
- Updated: more QUDA optimizations (reduced prec chrono vecs etc.): 392.6 sec
  - **10.2x walltime x 8-fold GPU reduction => 81.6x (Contribution to 2019 ECP FOM)**

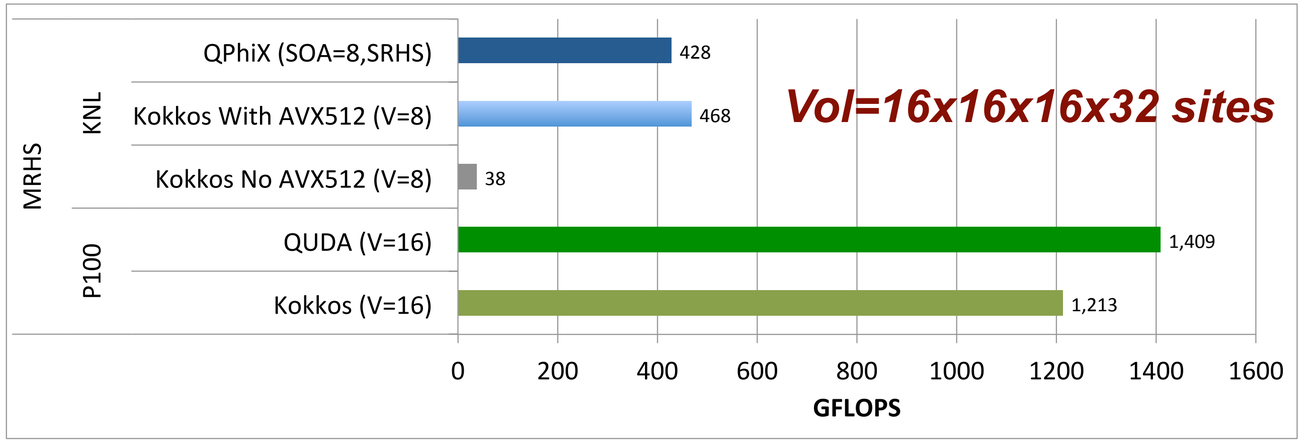
# Future Work/Challenges

- Raw Performance through Multi-RHS solvers
  - N-RHS at a time can in principle reap N-fold reuse of gauge field (*subject to other bottlenecks)
  - Interesting MRHS algorithms out there (e.g. using auxiliary RHS, or for changing matrices)
- Scaling Challenges
  - Usual MG scaling challenges (data parallel degrees of freedom get coarsened away)
  - Use domain decomposed preconditioners (?)
  - Multi-RHS can help with strong scaling in MG (more network friendly)
  - Multi-shift solvers for rational approximations in RHMC are still a challenge
- Better MD Integrators?
- More efficient Markov-Chain Monte Carlo methods ?  (also studied in LQCD ECP)
- Performance Portability (we are OK currently, but Aurora/Frontier will be v. new)

# Current Kokkos Performance Summary

- SRHS Case:
  - ***Kokkos Vectorized Dslash with AVX512 and tuned blocking matches QPhiX on Cori KNL node (68 cores, 272 threads)***
  - Unvectorized & No AVX cases are slow
  - Kokkos Naive CUDA version is 72% of QUDA on P100 (SummitDev)
  - ***Vectorized (but V=1) QUDA version benefits from block tuning, memory & locality optimizations and md_parallel_for: 79% of QUDA on P100 (SummitDev)***
- MRHS Case:
  - ***Kokkos With AVX512 exceeds corresp. QPhiX SRHS performance on Cori KNL node for 8 RHS***
  - Kokkos Without AVX512 is very slow
  - ***Kokkos CUDA version is 86% of QUDA for 16 RHS on SummitDev (P100)***



# Summary

- A variety of applied maths topics touched in LQCD
- Linear Solvers have traditionally received most focus
- Adaptive Multi-grid for Wilson Clover Fermions (developed thanks to SciDAC) has been a game changer for LQCD - other fermion actions in development
- Looking to the future we always need better solvers, better MD integrators and better Monte-Carlo methods
- A large focus with machines coming down the road is performance portability
  - Encouraging early Kokkos experiences. Would like to start looking at Kokkos SIMD types, and potentially performance portable Batched BLAS (Kokkos Kernels?) Also looking at SyCL and OpenMP-5 for future architectures (ECP).
- Our SciDAC project also has components looking at Optimization and Workflow.
- Questions?

# Acknowledgments


- Jefferson Lab is operated by Jefferson Science Associates LLC under U.S. DOE Contract No. DE-AC05-06OR23177
- B. Joo gratefully acknowledges funding from the U.S. Department of Energy, Office of Science, Offices of Nuclear Physics, High Energy Physics and Advanced Scientific Computing Research under the SciDAC programs: SciDAC, SciDAC 2, SciDAC 3 and SciDAC 4.
- B. Joo gratefully acknowledges funding from the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under the USQCD Exascale Computing Project (Lattice QCD)
- B. Joo gratefuly acknowledges travel funding from NERSC for a summer Affiliate Appointment for work on Kokkos.
- The 2017 ORNL Hackathon at NASA was a collaboration between and used resources of both the National Aeronautics and Space Administration and the Oak Ridge Leadership Computing Facility at Oak Ridge National Laboratory.  Oak Ridge Nation Laboratory is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.
- We gratefully acknowledge use of computer time at JeffersonLab (SciPhi XVI cluster), K80 Development node, NERSC Cori and OLCF SummitDev.